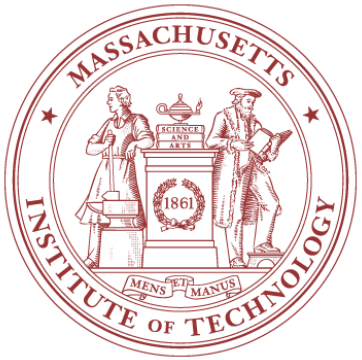
Viterbi Algorithm - Implementation

Lecture 14

Vladimir Stojanović

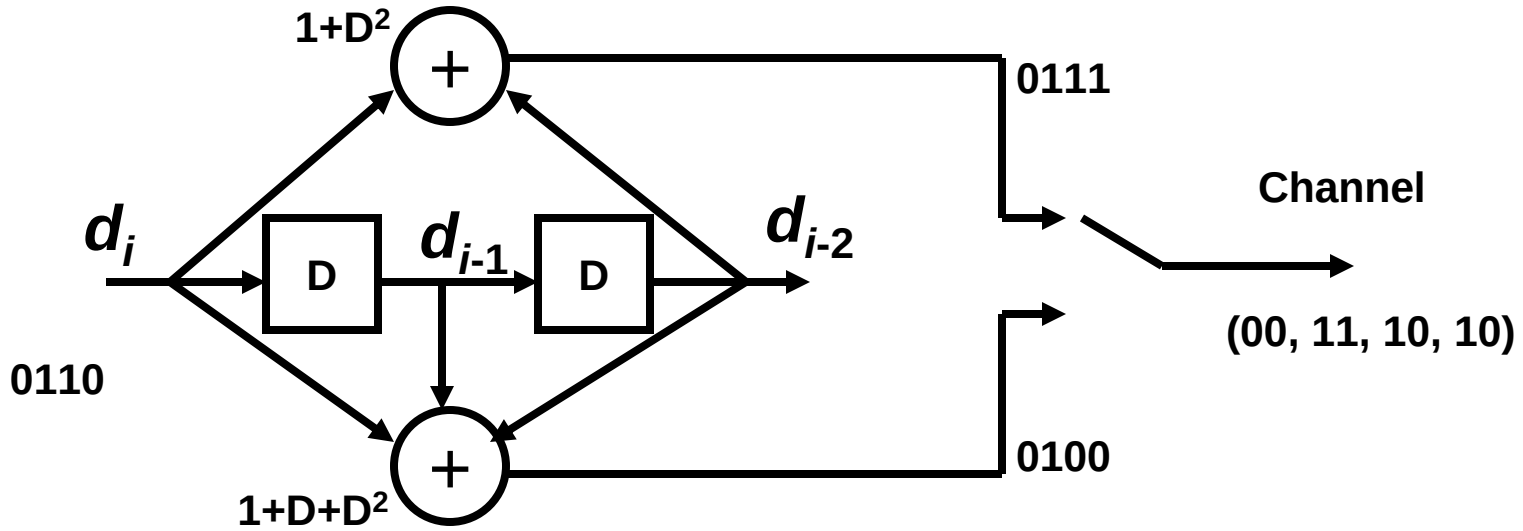


6.973 Communication System Design – Spring 2006

Massachusetts Institute of Technology

Convolutional Codes

- Adding redundancy

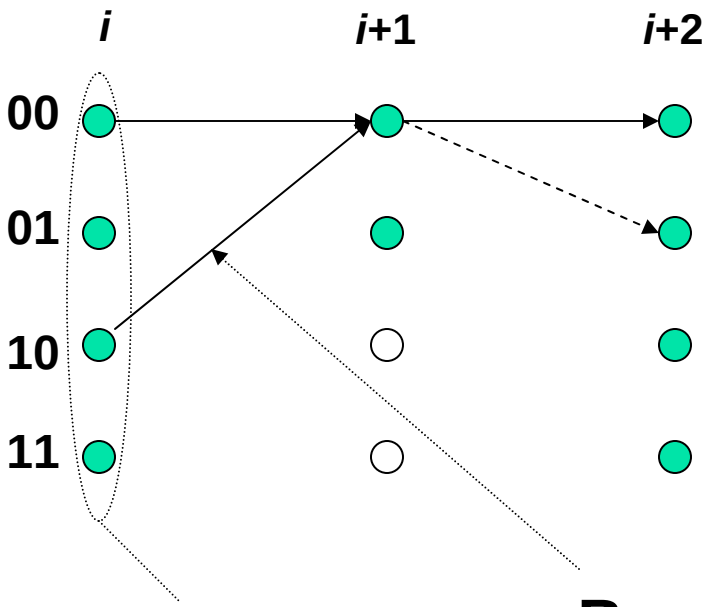


- Generators:

$G_1 = 101$

$G_2 = 111$

Decisions at Each Step



State (path) metrics

Branch metrics

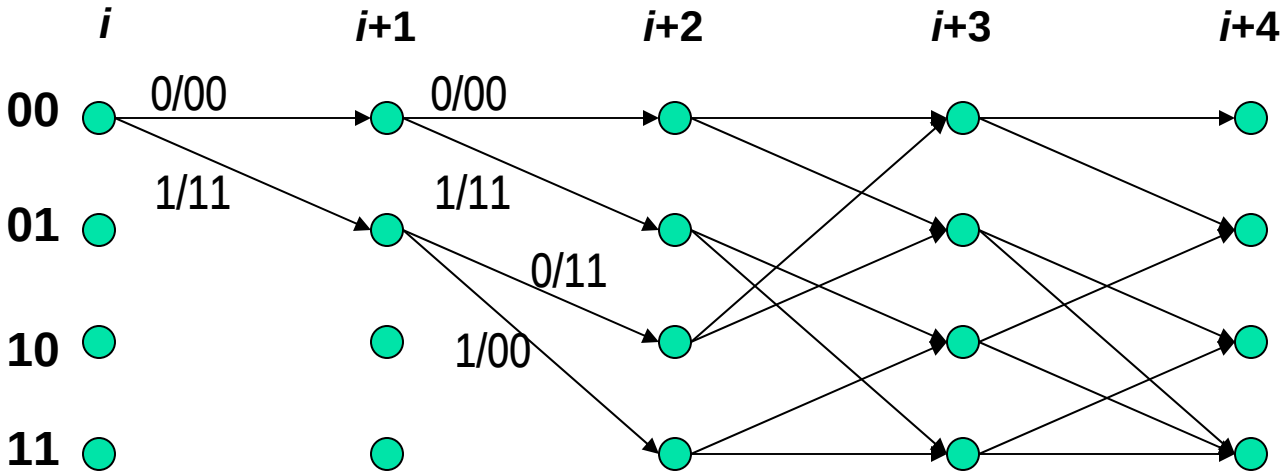
- In Gaussian channel:
  $bm = (y_k - s_k)^2$
- In BSC:
  $bm = d_H(y_k, s_k) = |y_k - s_k|$

d_H is Hamming distance

- If received $y_k = 10$
  $d_H(10, 00) = 1$
  $d_H(10, 01) = 2$

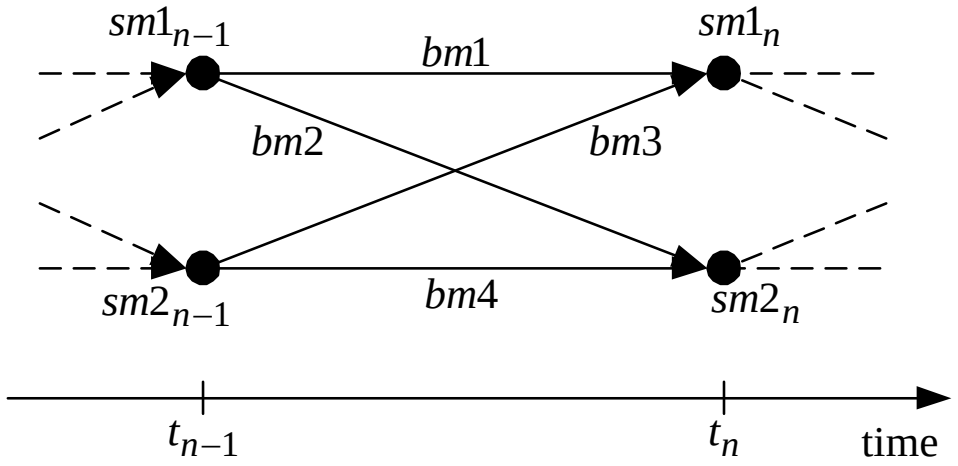
Trellis Diagram

- Time-indexed state diagram



The Viterbi Algorithm

- Illustrated by 2-state trellis



$$sm1_n = \min\left(sm1_{n-1} + bm1,\ sm2_{n-1} + bm3\right)$$

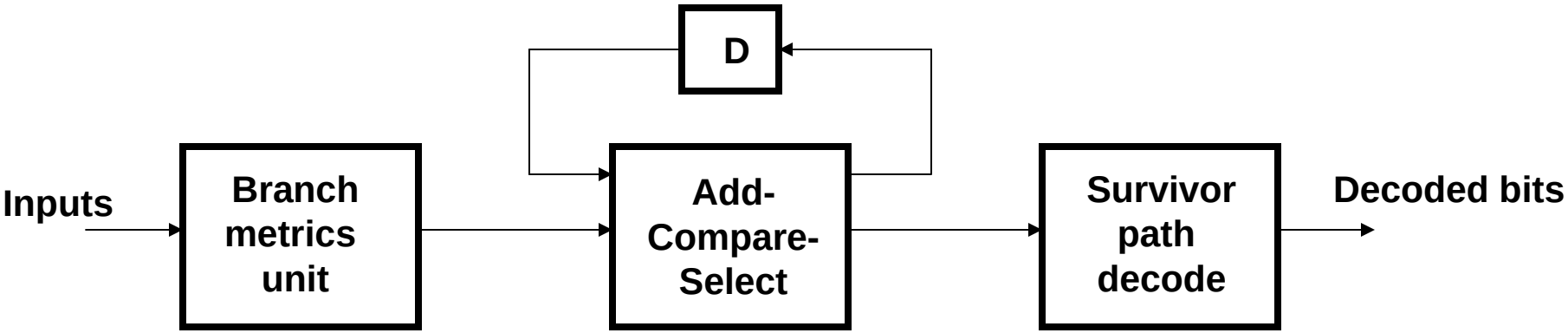
$$sm2_n = \min\left(sm1_{n-1} + bm2,\ sm2_{n-1} + bm4\right)$$

Add Add Select Compare

Viterbi Decoder

- Implements Viterbi algorithm
- Three main components
  - Branch metric calculation
  - Path metric accumulation (add-compare-select recursion)
  - Survivor path decode

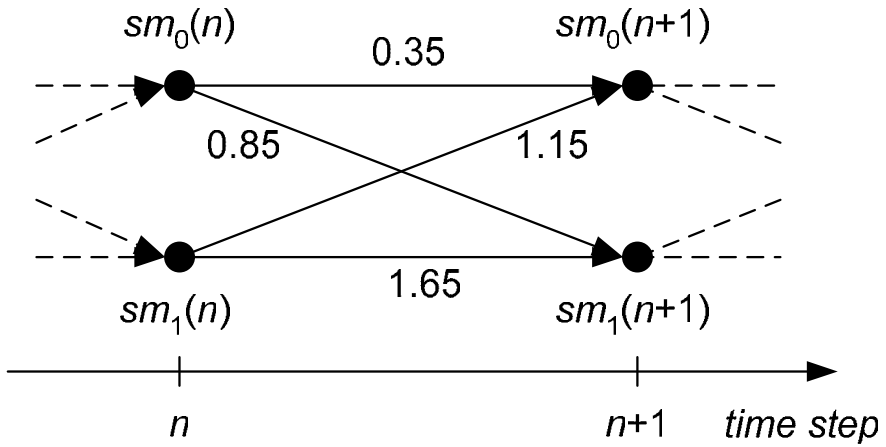
Viterbi Decoder



- Branch metrics unit
  - Calculates the distances between the received signal and the ideal signals
- Add-Compare-Select unit
  - Accumulates path metrics
- Survivor path decode
  - Keeps track of the path through the trellis

Calculating Branch Metrics

- Assume G_1 and G_2 both output a 0
- Soft decoder inputs are 0.1 and 0.25 instead
- Branch metrics:
  - $bm_{00} = |0 - 0.1| + |0 - 0.25| = 0.35$
  - $bm_{01} = |0 - 0.1| + |1 - 0.25| = 0.85$
  - $bm_{10} = |1 - 0.1| + |0 - 0.25| = 1.15$
  - $bm_{11} = |1 - 0.1| + |1 - 0.25| = 1.65$

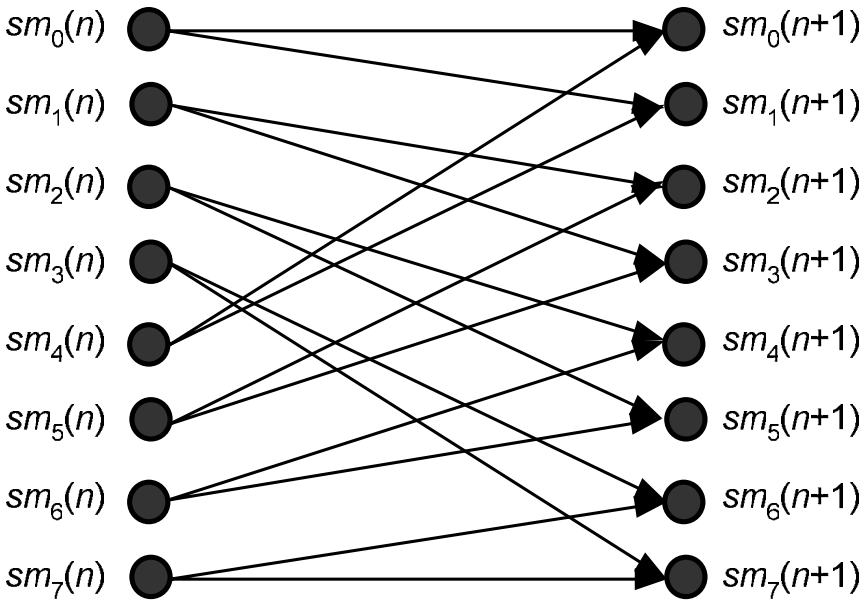


Calculating Branch Metrics

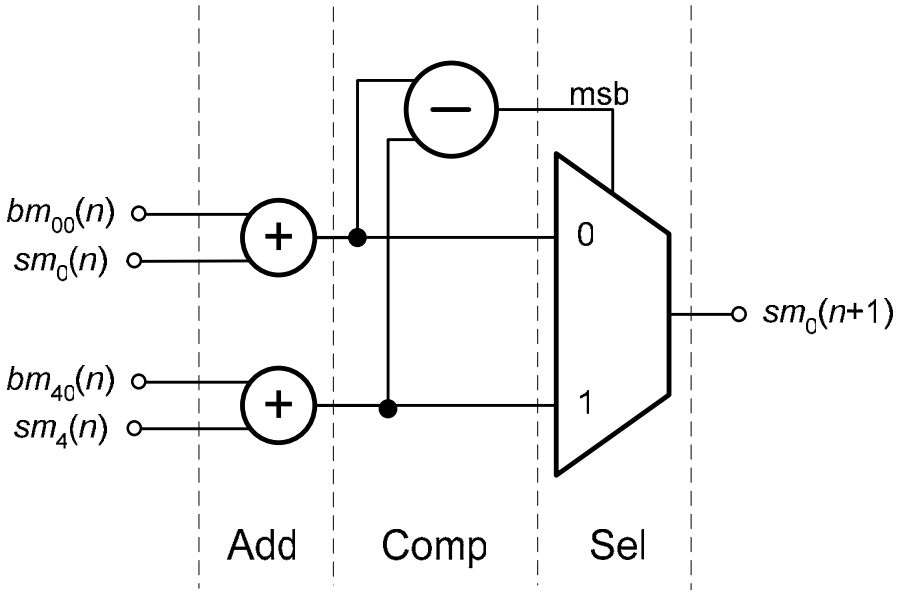
- Euclidean distances (AWGN channel)
- $bm_i = (y_k - s_k)^2$
- s_k are usually integers

  $bm_i = y_k^2 - 2C_i y_k + C_i^2$
- Since y_k terms are common to all branch metrics they drop out in ACS comparison, and can be eliminated.
- C_i^2 are precomputed and $2C_i y_k$ are shifts and adds.

Add-Compare-Select Recursion

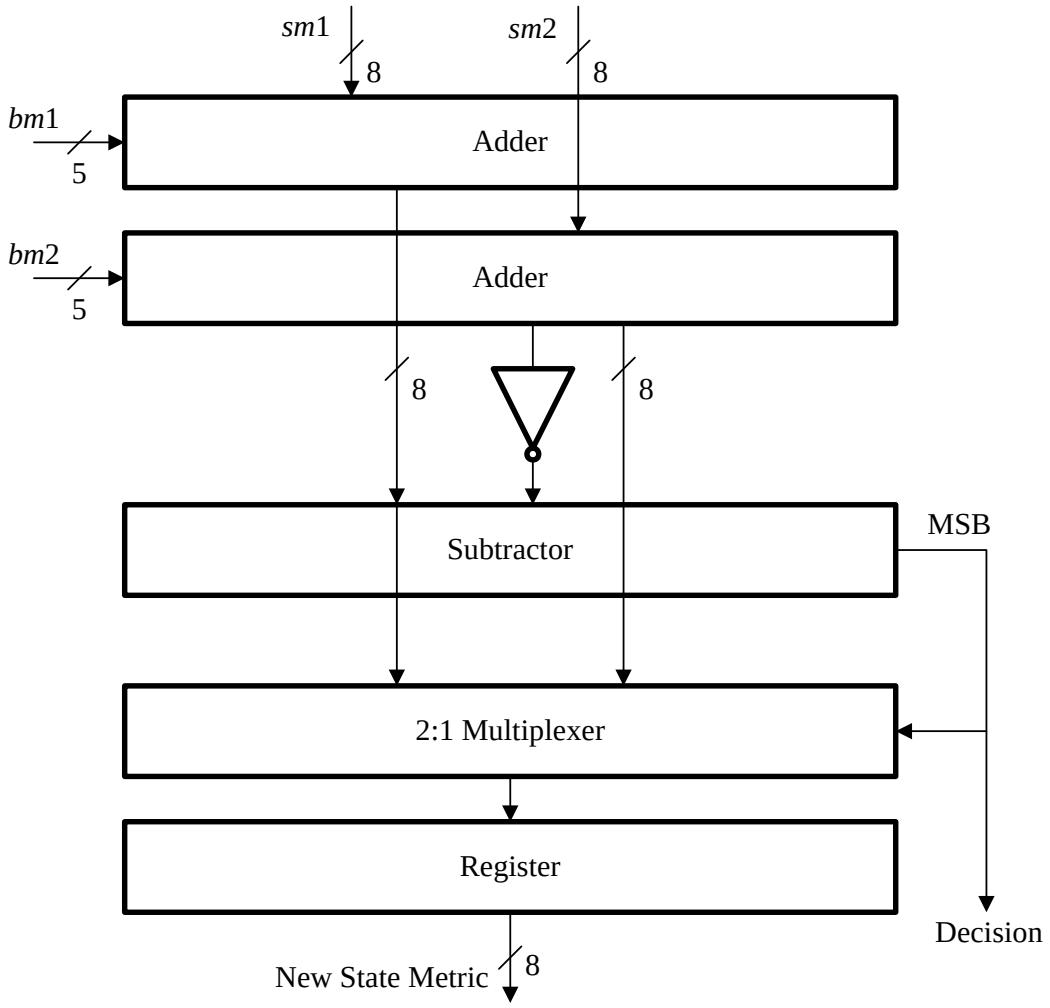


Eight state trellis

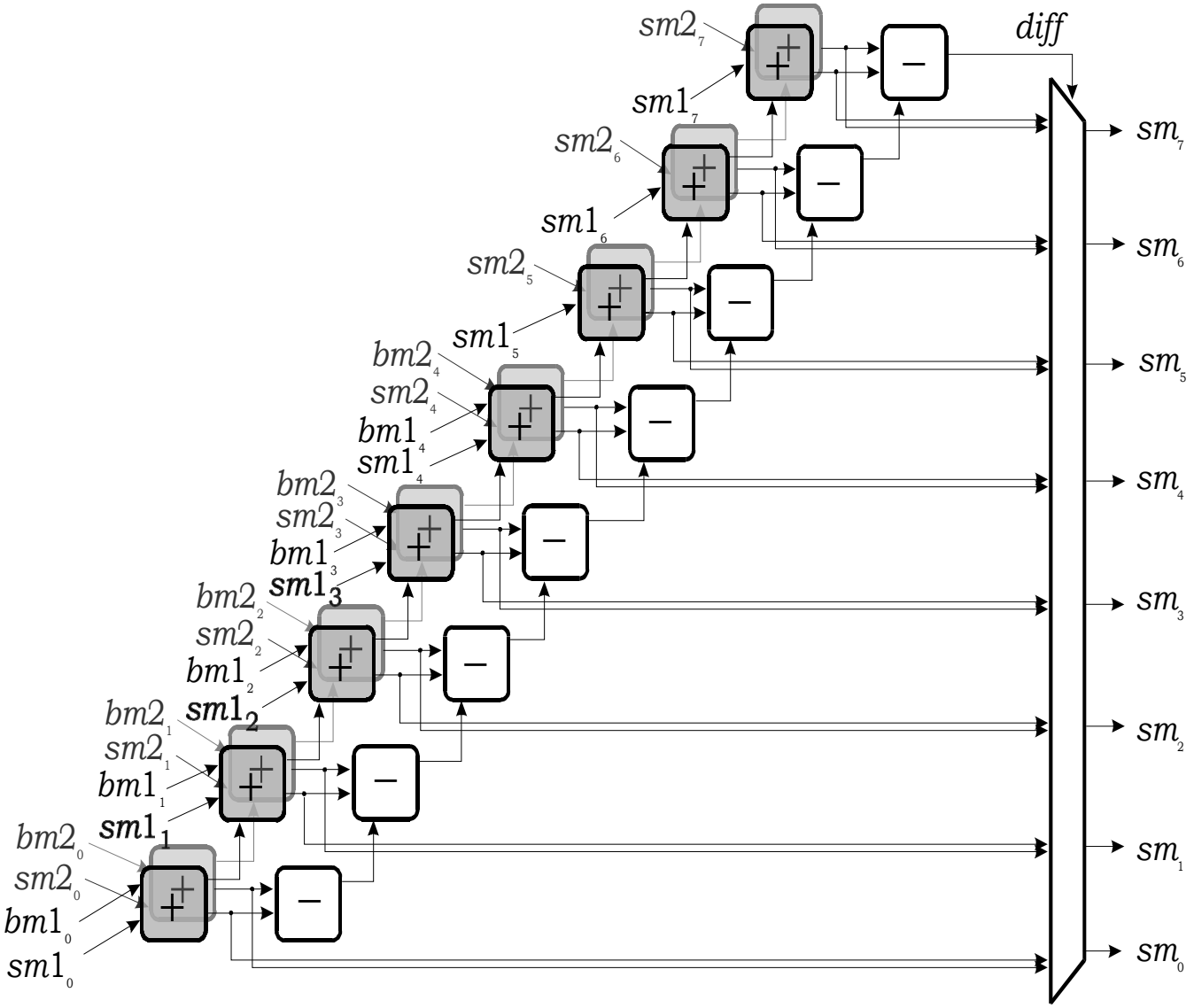


Conventional
add-compare-select unit

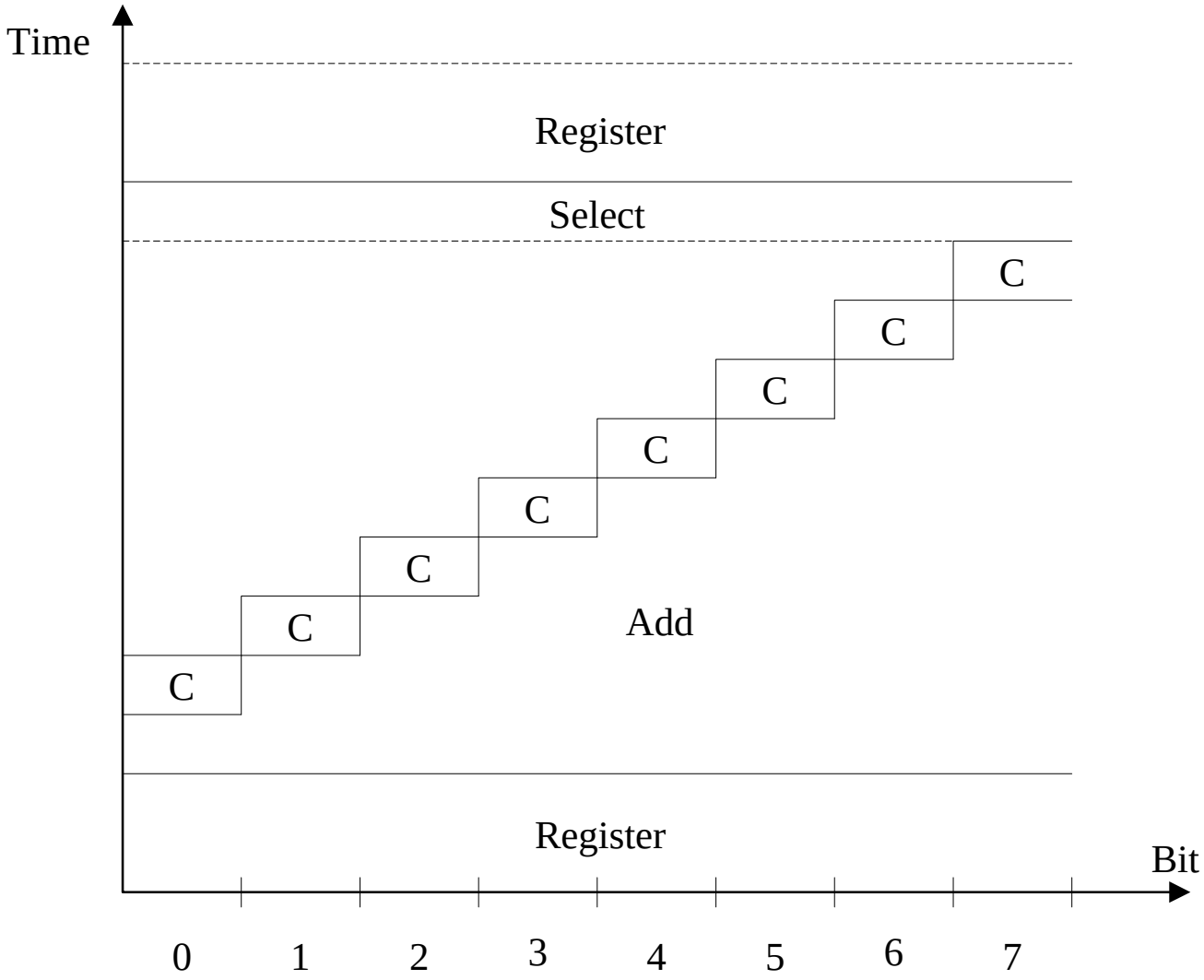
Add-Compare-Select Recursion



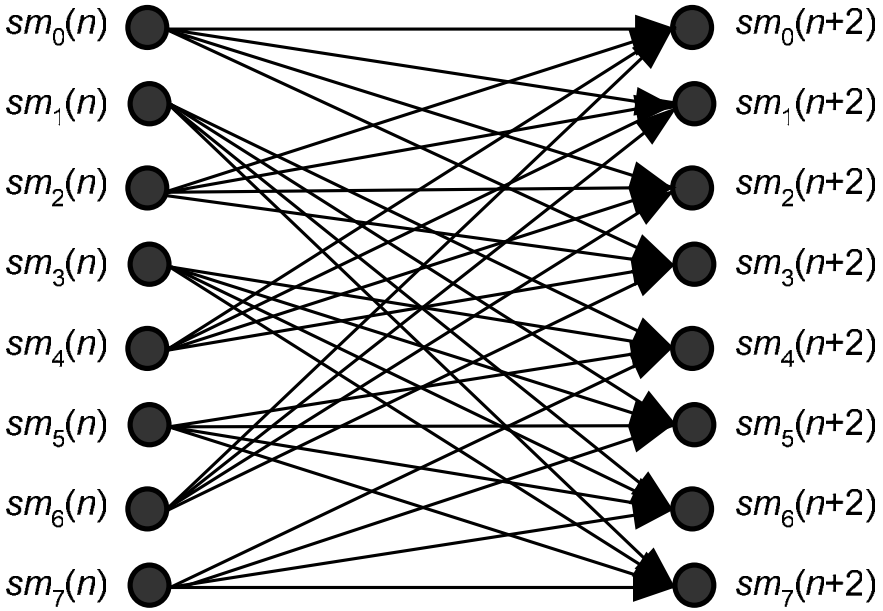
Add-Compare-Select Recursion



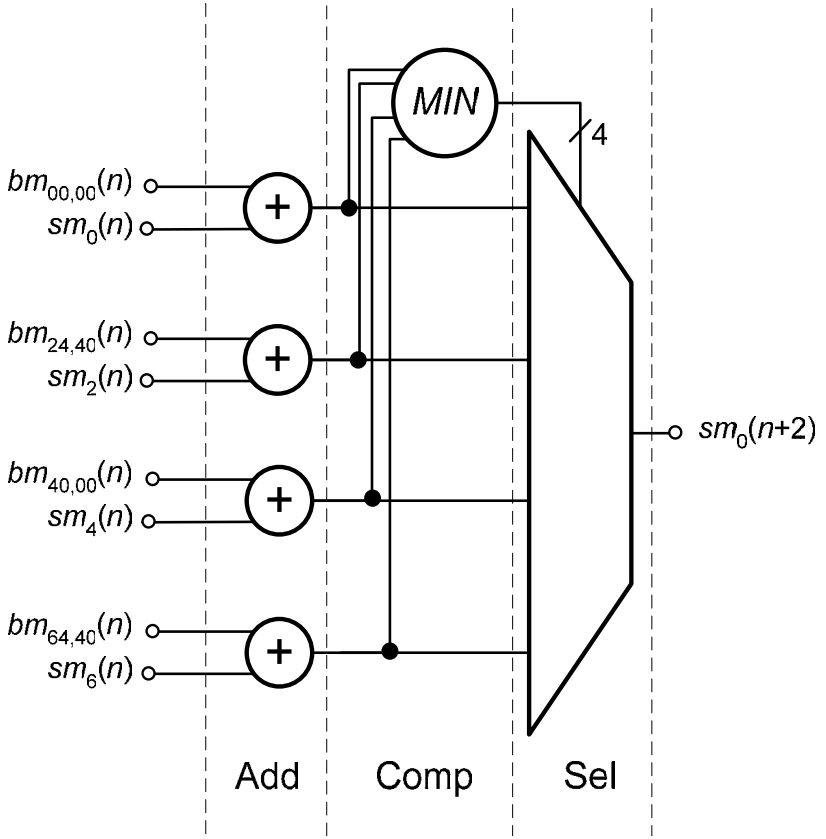
Add-Compare-Select Recursion



Add-Compare-Select Recursion



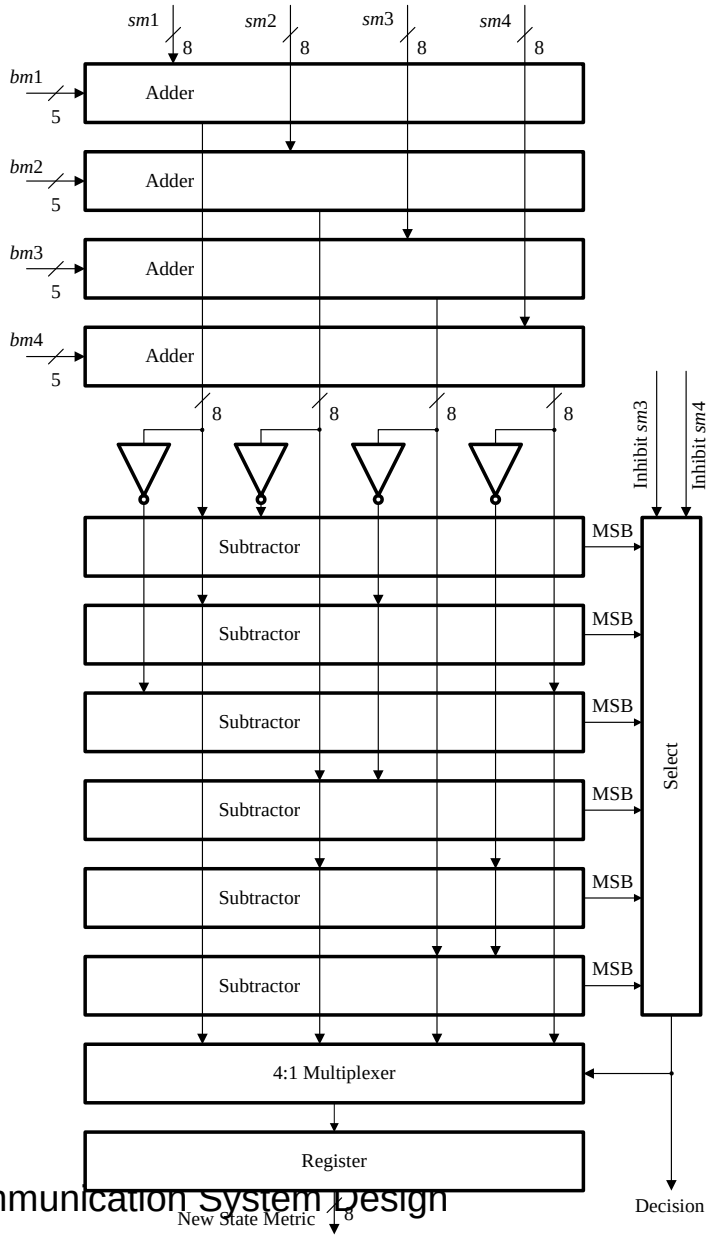
One step lookahead
applied to an eight-state trellis



Radix-4
add-compare-select unit

Add-Compare-Select Recursion

4-way ACS:
2 additions +
6 comparisons



2-state example

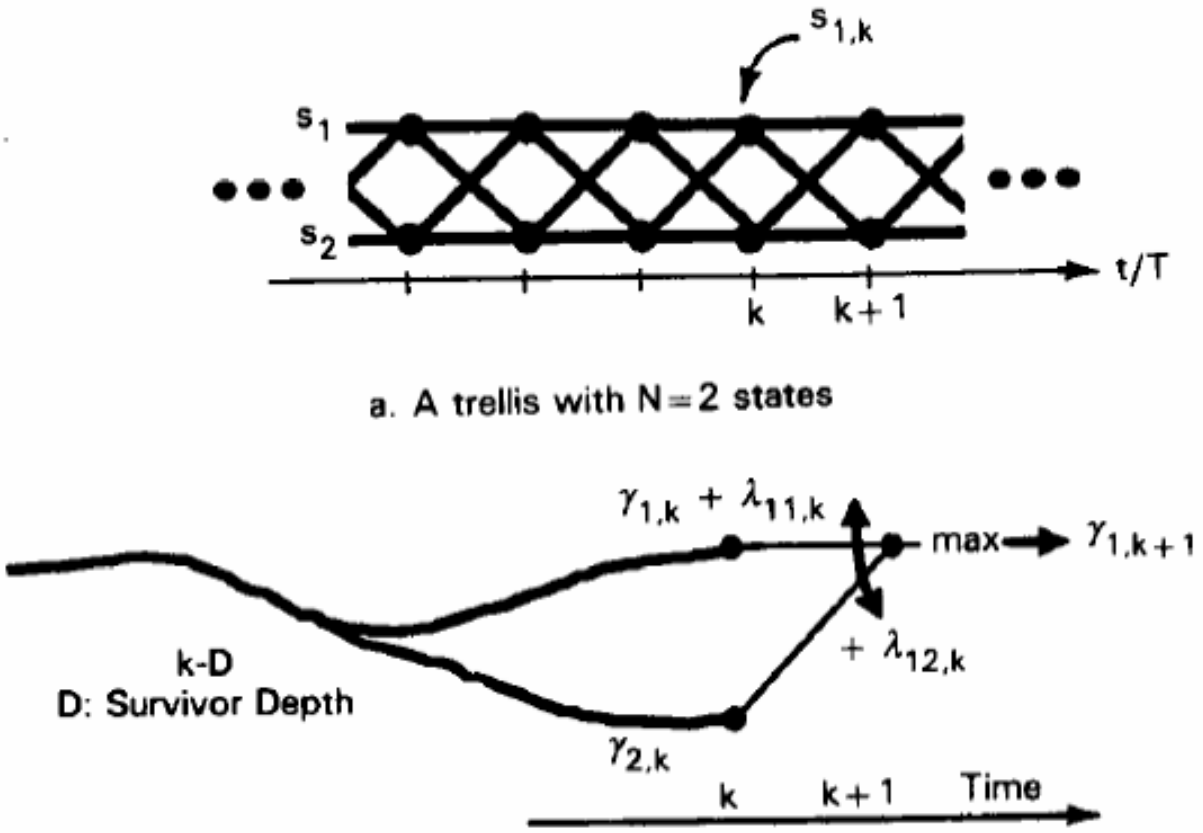


Figure from Fettweis, G., and H. Meyr. "High-speed Parallel Viterbi Decoding: Algorithm and VLSI-architecture." *IEEE Communications Magazine* 29 (1991): 46-55. Copyright 1991 IEEE. Used with permission.

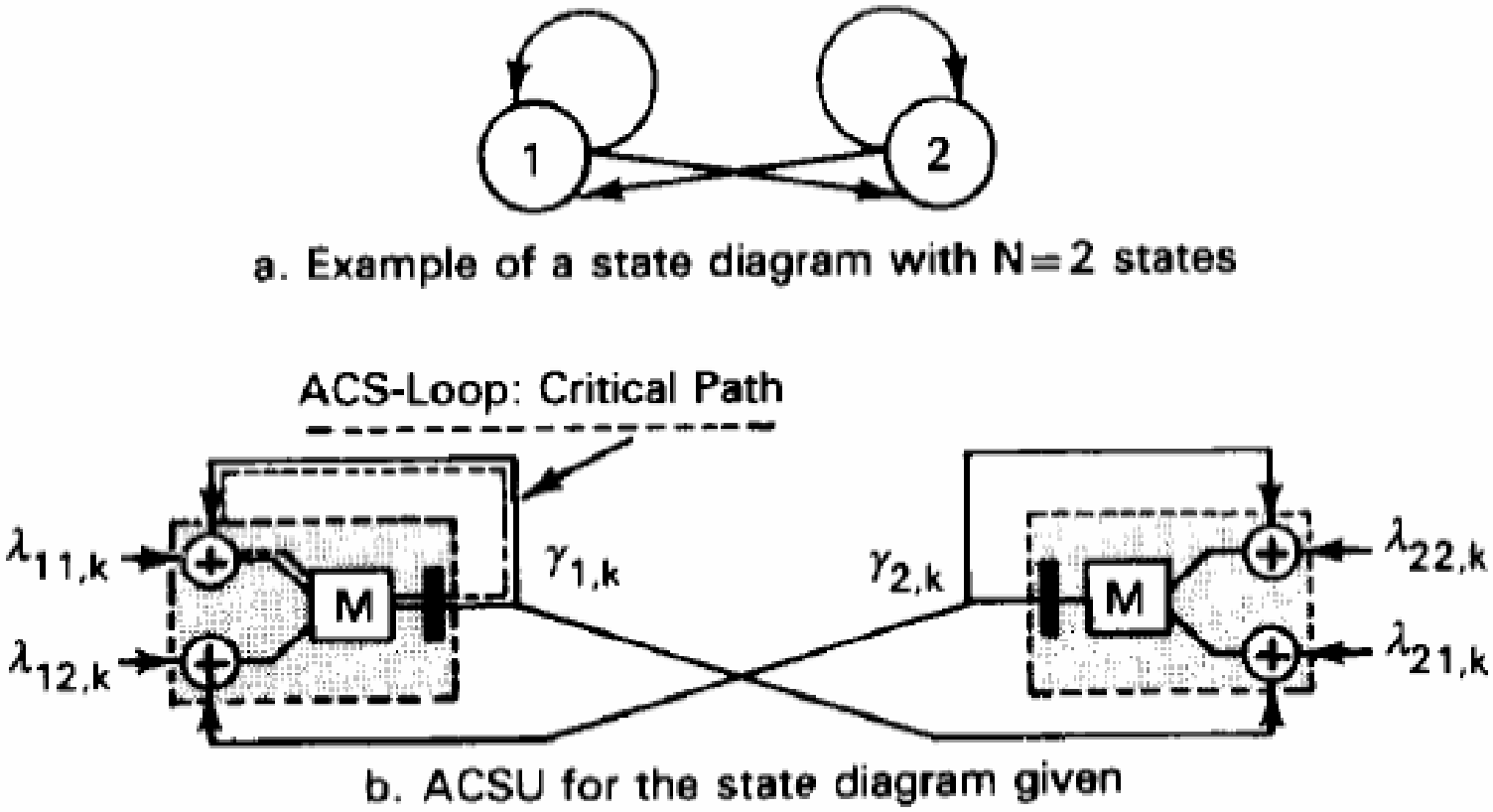
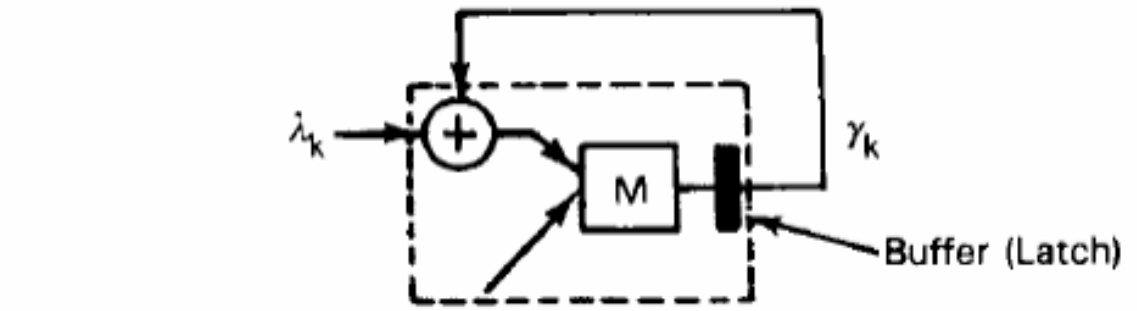
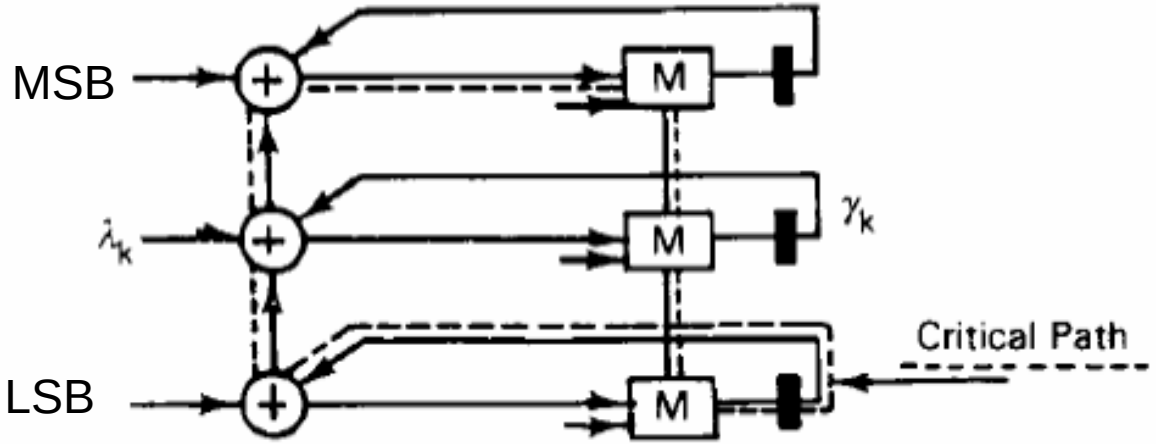


Figure from Fettweis, G., and H. Meyr. "High-speed Parallel Viterbi Decoding: Algorithm and VLSI-architecture." *IEEE Communications Magazine* 29 (1991): 46-55. Copyright 1991 IEEE. Used with permission.

Bit-level view



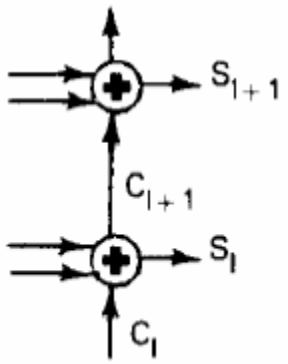
a. The ACS-loop which comprises an addition and a maximum selection (M).



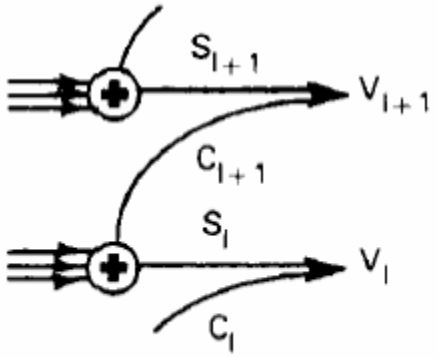
b. A 3-bit implementation of the ACS-loop.

Figure from Fettweis, G., and H. Meyr. "High-speed Parallel Viterbi Decoding: Algorithm and VLSI-architecture." *IEEE Communications Magazine* 29 (1991): 46-55. Copyright 1991 IEEE. Used with permission.

Turn into forward path and pipeline



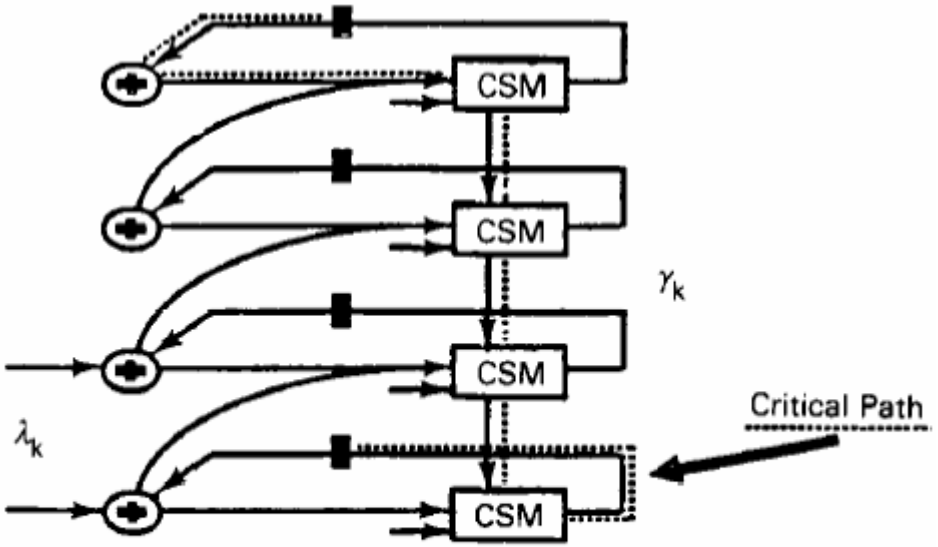
a. Carry-ripple addition.



b. Carry-save addition.

$$S = \sum_i (s_i + c_i) 2^i = \sum_i v_i 2^i$$

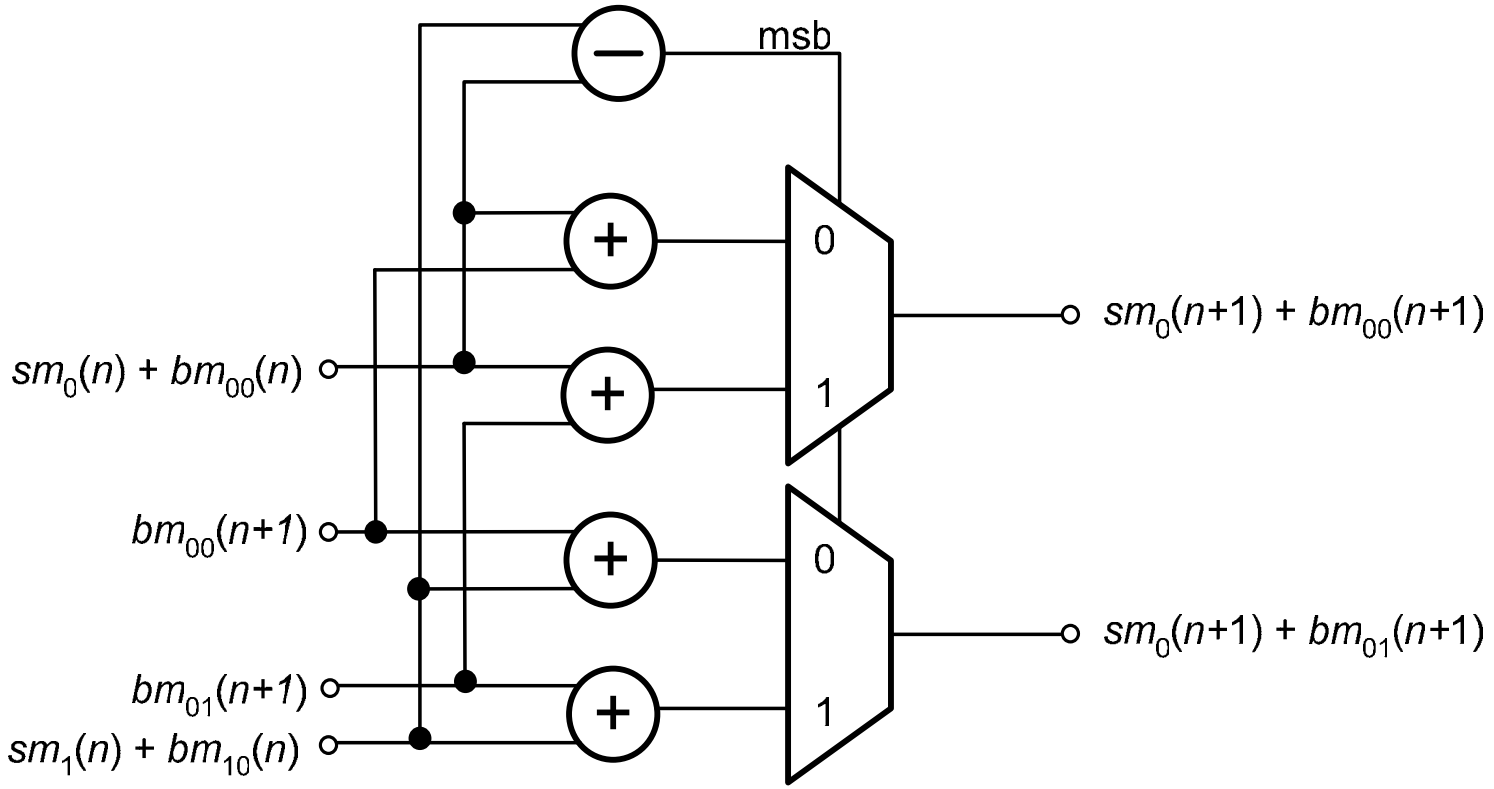
$$S = \sum_i s_i 2^i$$



4-bit ACS-loop with CS-arithmetic.

Figures from Fettweis, G., and H. Meyr. "High-speed Parallel Viterbi Decoding: Algorithm and VLSI-architecture." *IEEE Communications Magazine* 29 (1991): 46-55. Copyright 1991 IEEE. Used with permission.

Add-Compare-Select Recursion



Parallel compare-select-add unit

References

- Slides from Borivoje Nikolic and Bob Brodersen
- G. Fettweis and H. Meyr "High-speed parallel Viterbi decoding: algorithm and VLSI-architecture," *Communications Magazine, IEEE* vol. 29, no. 5, pp. 46-55, 1991.